

DTIC FILE COPY

4

RADC-TR-87-250

Final Technical Report

March 1988



AD-A196 630

INTEGRATED EXPERT SYSTEMS

SCEEE

Stuart H. Hirshfield

DTIC
ELECTE
MAY 24 1988
S & D

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

**ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, NY 13441-5700**

90 23 270

This report has been reviewed by the RADC Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

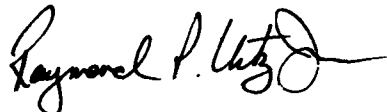
RADC-TR-87-250 has been reviewed and is approved for publication.

APPROVED:



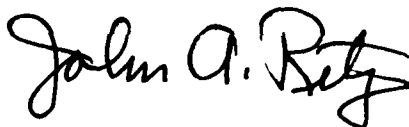
YALE SMITH
Project Engineer

APPROVED:



RAYMOND P. URTZ, JR.
Technical Director
Directorate of Command & Control

FOR THE COMMANDER:



JOHN A RITZ
Directorate of Plans & Programs

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC (COAD) Griffiss AFB NY 13441-5700. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 0704-0188	
1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED			1b. RESTRICTIVE MARKINGS N/A		
2a. SECURITY CLASSIFICATION AUTHORITY N/A			3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution unlimited.		
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE N/A					
4. PERFORMING ORGANIZATION REPORT NUMBER(S) N/A			5. MONITORING ORGANIZATION REPORT NUMBER(S) RADC-TR-87-250		
6a. NAME OF PERFORMING ORGANIZATION SCEE		6b. OFFICE SYMBOL (if applicable)	7a. NAME OF MONITORING ORGANIZATION Rome Air Development Center (COAD)		
6c. ADDRESS (City, State, and ZIP Code) 1101 Massachusetts Ave St Cloud FL 32769			7b. ADDRESS (City, State, and ZIP Code) Griffiss AFB NY 13441-5700		
8a. NAME OF FUNDING / SPONSORING ORGANIZATION Rome Air Development Center		8b. OFFICE SYMBOL (if applicable) COAD	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F30602-81-C-0193		
8c. ADDRESS (City, State, and ZIP Code) Griffiss AFB NY 13441-5700			10. SOURCE OF FUNDING NUMBERS		
			PROGRAM ELEMENT NO. 62702F	PROJECT NO. 5581	TASK NO. 24
					WORK UNIT ACCESSION NO. 44
11. TITLE (Include Security Classification) INTEGRATED EXPERT SYSTEMS					
12. PERSONAL AUTHOR(S) Stuart H. Hirshfield					
13a. TYPE OF REPORT Final		13b. TIME COVERED FROM Jun 86 to Jun 87		14. DATE OF REPORT (Year, Month, Day) March 1988	
15. PAGE COUNT 24					
16. SUPPLEMENTARY NOTATION N/A					
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)		
FIELD	GROUP	SUB-GROUP			
12	05	09			
12	05	04	Expert Systems, Cooperating Expert Systems, Distributed Problem Solving		
19. ABSTRACT (Continue on reverse if necessary and identify by block number) This report presents an analysis of the current theoretical and practical states of what has been dubbed "cooperating expert systems (COPEs) technology" - i.e. studies and experiments devoted to the problem of coordinating the communication and problem-solving activities of distinct expert (and non-expert) systems that already exist and that were developed independently. This problem is seen as distinct from those addressed by a number of more general technical approaches to achieving cooperation. The approaches, as represented by the black-board architecture, the Airland Loosely Integrated Expert Systems (ALLIES) effort, the A Better Environment (ABE) effort, and the Multiple Node Expert System project, are described and evaluated in terms of their applicability to the existing system problem. A software architecture is proposed which combines aspects of these approaches to address a variety of practical tradeoffs that arise. Finally, a design technique that seems particularly suited to the architecture is described.					
20. DISTRIBUTION / AVAILABILITY OF ABSTRACT <input checked="" type="checkbox"/> UNCLASSIFIED/UNLIMITED <input type="checkbox"/> SAME AS RPT. <input type="checkbox"/> DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED		
22a. NAME OF RESPONSIBLE INDIVIDUAL YALE SMITH			22b. TELEPHONE (Include Area Code) (315) 330-7764		22c. OFFICE SYMBOL RADC (COAD)

DD Form 1473, JUN 86

Previous editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

UNCLASSIFIED

UNCLASSIFIED



Accession For	
NTIS CRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Date	
Availability Codes	
Dist	Avail and/or Special
A-1	

Cooperative Expert Systems: A Preliminary Analysis

Stuart H. Hirshfield

1.0 Introduction

The Cooperative Expert Systems (COPES) effort currently being considered by RADC/COAD has as its main objective the development of design criteria and a prototype design for a system to be composed in large part from existing expert systems. Three expert systems, the Automated C³CM Battle Management Decision Aid (ACBMDA), the Tactical Expert Mission Planner (TEMPLAR) and FORCES, have been recognized as having a potential for cooperation in the sense that they can, in theory, provide useful information and decision-making support to one another. A separate, knowledge-based expert, the Cooperative Expert System is to be designed to arbitrate and coordinate this cooperation. It is anticipated that the prototype design will illuminate general design principles which, in turn, address the practical tradeoffs involved in building such systems.

Preliminary to the prototype design, the COPES effort provides for (1) an analysis of the state of the art in cooperating systems technology, and (2) the compilation of a comprehensive bibliography. Four approaches, as represented by the *blackboard architecture*, MITRE's *Airland Loosely Integrated Systems (ALLIES)* effort, Teknowledge's *A Better Environment (ABE)* effort, and the *Multiple Node Expert System* under development at the US Army Intelligence Center and School, are mentioned explicitly as candidates for analysis. The purpose of this paper is to provide background materials in support of these preliminary steps.

Specifically, section 2 of this paper presents an overview of the issues to be faced in designing cooperative systems and illustrates the distinguishing features of the COPES environment. The following four sub-sections (3.1 - 3.4) describe, respectively, the aforementioned approaches to achieving cooperation in terms of their suitability for addressing COPES scenarios. Advantages, disadvantages and representative applications are presented for each approach. Tentative conclusions and recommendations, as embodied by a working design (which incorporates various features from each of the technical approaches evaluated), an accompanying design technique, and a brief example, are presented in section 4. Section 5 represents a first

COPES: A Preliminary Analysis

crack at the bibliography.

2.0 Theoretical Background

As expert systems proliferate and as their domains of expertise begin to interrelate, there is increasing interest in combining existing systems into larger, potentially more powerful tools. Existing systems which share some high-level goal have been combined, as have those which share low-level software utilities and/or hardware resources. As the independent systems involved become increasingly "knowledge-based" (i.e., self-aware), it becomes realistic to imagine groups of them *which cooperate with one another in more sophisticated ways*. If, for example, they were aware of one another's capabilities, goals and activities, they may be able to support one another to the extent of influencing each other's actions and decisions. The blackboard and distributed problem solving architectures espoused in the literature of distributed artificial intelligence (DAI) are intended to facilitate such cooperation. Implementations based on these architectures typically describe systems comprised of independent agents that are coordinated from the outset in the sense that they have access to common data areas and to uniform communication facilities. The COPES effort is intended to address the problems and tradeoffs one faces in attempting to achieve comparable levels of cooperation between already existing and operational systems about which we can make few if any assumptions. At its highest level, COPES can thus be regarded as a feasibility study.

A variety of problem solving architectures which are intended to facilitate cooperation among collections of decentralized, loosely coupled agents have been described in the DAI literature. Most are based, either implicitly or explicitly, on the demonstrated utility of a blackboard-like architecture, which provides a global data base for the posting and communication of intermediate results as produced by independent processing elements all of which are contributing to the solution of a common problem. Control over the processing elements and of the data base itself are typically handled by a separate agent(s) which embodies a global problem solving algorithm. More recently, networks of problem solving agents have been described which use local and global blackboards to accomplish cooperation among a number of relatively independent ongoing tasks. "Local controllers" or "meta experts" are incorporated to serve as extended interfaces for the individual agents. Such interfaces typically describe for each agent its relation to the general problem being solved and to the other agents in the system both in terms of the data and the tasks that are being shared.

Some of the problems addressed by these architectures, e.g., the partitioning and assignment of sub-tasks to individual agents, are moot when one is constrained to working with predefined experts. Problems of agents working at cross-purposes are minimized, as is the likelihood of redundancy or duplicated effort among nodes. Unfortunately, other problems of a more practical, software engineering nature replace these.

Following the principle "If it ain't broke, don't fix it", we have established as a primary criteria for a suitable cooperative framework that it impose a minimum of

COPES: A Preliminary Analysis

change on the expert systems being coordinated. In short, we do not want to reimplement the existing experts. Rather, we want as much as possible to take advantage of the modularity imposed on the global problem by having already available agents for accomplishing well defined sub-tasks. If such agents have to be modified (as they undoubtedly will), we would like changes to be localized and controlled. This criteria is formidable when coordinating any programs, but is particularly so when the systems involved are "expert". Such systems are by definition highly-specialized, knowledge-intensive programs with complex control schemes that are often difficult to modify or extend to other domains. Furthermore, the blackboard and network DAI architectures require of their distinct agents, at a minimum, an awareness of the global data area and, more often, extensive knowledge of both the global problem and the other agents involved. The trend in such architectures has been toward increasingly sophisticated local controllers which serve to adjust the behavior of the individual agents based on both global constraints and local constraints involving other agents.

Secondarily, an idealized framework would make a minimum of assumptions about the agents being coordinated. It is perfectly realistic, for example, to imagine an expert system cooperating with a database system which is not expert in the sense of having either explicit control over or an awareness of its actions. It would be presumptuous to expect that every independent agent would be compatible, say, with a blackboard architecture, and it would be unrealistic to assume that all agents involved could be readily modified so as to be compatible with any chosen architecture.

3.0 Technical Approaches

It is against this backdrop that we consider four established approaches to achieving cooperation. Each makes different assumptions about the agents whose activities are to be coordinated and about the global problem being addressed. Each achieves a different level of cooperation at a different expense, reflected most directly in the degree of coupling imposed on the individual agents. All are alike, though, in two critical ways: (1) they have demonstrated utility, and (2) they embody general design criteria of theoretical interest to the COPES effort. [Note: It is not my intention here to describe these approaches in any technical detail, as has been done in various of the cited references. I will instead summarize each approach only to the extent that it can be evaluated in terms of its relevance to COPES.]

3.1 Blackboard Architectures

The blackboard model has evolved as a result of abstracting the architectural features of the HEARSAY-II speech understanding system into a general framework for cooperative problem solving. A variety of expert systems have since been implemented using a variety of interpretations of the basic blackboard architecture. All interpretations can be described in terms of multiple knowledge sources communicating implicitly through and responding opportunistically to changes in a global data base called the "blackboard".

COPES: A Preliminary Analysis

Knowledge sources are regarded as separate and independent sub-domain experts, each of which can contribute to the solution of a single, more general problem. The blackboard is the sole means of communication between the knowledge sources. Each knowledge source is responsible for recognizing conditions on the blackboard to which it can respond, and for posting its intermediate results and hypotheses on the blackboard.

The blackboard itself is a highly structured collection of computational and solution-space information needed by the knowledge sources. It is typically organized hierarchically to reflect distinct, but related levels of analysis. Knowledge sources may be similarly organized in the sense that each may be expert at a particular level of analysis - i.e., a knowledge source is often described as processing information at one level of analysis and producing information useful at other levels.

While knowledge sources are self-selecting, a control agent is responsible for monitoring changes to the blackboard and for choosing among available sources of activity. Based on the current condition of the blackboard and a global notion of interest or purpose, the control agent focusses its attention on either the blackboard or on a particular knowledge source. The iterative control cycle can be summarized as: (1) modify the blackboard as prescribed by a knowledge source, (2) poll the knowledge sources, (3) choose a focus of attention, and (4) activate the appropriate knowledge source. Problem-dependent criteria are used for determining when to terminate the cycle.

The most well-known of the applications which make use of this framework are the HEARSAY efforts at speech understanding, the HASP system for ocean surveillance, and, more recently, the CAGE and POLIGON systems for exploiting parallelism in blackboard models. It is not an overstatement to say that most of what we now refer to as "distributed artificial intelligence" has its seeds in the original work that led up to the HEARSAY project and in the original blackboard model. This is certainly true of the COPES effort.

Obviously, for independent agents to contribute to one another's problem solving activities, they must have a means for sharing "real-time information" - i.e., describing their processing goals, hypotheses and intermediate results. The idea of posting such information in a globally accessible data base is seemingly preferable to direct communication between agents, particularly when one considers the number of agents that may be involved in COPES-like cooperation and the difficulty in achieving direct communication between existing and potentially dissimilar programs and computers.

Opportunistic processing would also seem to be suited to a COPES scenario. In the interest of autonomy, one agent should not have to summon another explicitly when assistance is needed. This would entail each agent knowing which other agents are available for assistance and, worse still, how to invoke its assistants when they may (1) reside on other machines, (2) be written in other languages, and (3) have radically different knowledge representation schemes. If, on the other hand, each agent was privy to information that would allow it to decide independently when it was capable of contributing to an overall solution, the agents would be less tightly coupled

COPES: A Preliminary Analysis

and, in theory, would require less modification.

Less suited to COPES are the implied constraints that a blackboard approach imposes on the problem domain. Successful applications of blackboards have most often been to single, hierarchically-organized problems - i.e., ones in which the processing of each individual knowledge source is contributing to the solution of a well-defined, global problem at a predetermined level of analysis. Knowledge sources are independent in the sense that they have different levels of expertise within a single problem domain. It is more realistic to view the independent agents in a COPES scenario as having wholly different areas of expertise that happen to interrelate. If their relationship happened to be hierarchical, one agent (the lower-level one) could simply be subsumed by the other (the higher-level one) and be treated much like a subroutine. The far more interesting (and more relevant to COPES) case is that in which the agents are mutually supportive of one another - i.e., they are siblings. Such situations do not serve to structure the information on the blackboard in a useful way.

While opportunistic processing is intuitively more pleasing than is direct invocation between agents, it still requires of the independent knowledge sources that they monitor the blackboard individually for conditions to which they can respond. In a COPES scenario, when the agents involved have been implemented independently with, potentially, no knowledge of a blackboard, of the other agents they are to cooperate with, or of a global problem to which they are contributing, pure self selection could impose significant modifications on the previously independent agents.

3.2 Mitre's ALLIES

The ALLIES project represents an attempt to integrate two previously-developed knowledge-based tools, a mission planner (OPLANNER) and an intelligence data analyst (ANALYST). The two systems interact directly with one another and indirectly through a simulated battlefield environment model (BEM). Given an original operational goal, an initial analysis of the enemy situation as provided by ANALYST, and a description of the current state of friendly forces as found in the BEM, OPLANNER develops a mission plan and conveys it to the BEM. The BEM is updated to reflect the course of action prescribed by the plan. ANALYST both receives reports from friendly sensors as to the state of the BEM, and responds to specific OPLANNER requests for updated analysis of critical mission elements. OPLANNER compares all information received with its expectations and, ultimately, will be able to replan based on observed discrepancies.

In terms of straightforward cooperation, this study has little to offer COPES. In fact, the OPLANNER and ANALYST systems are written in the same language, run on identical machines, and communicate directly with one another via input/output streams (communication with the BEM is achieved using files). ANALYST can almost be regarded as a subprocess (which admittedly can be run on a standalone basis) of OPLANNER, which invokes it as it deems necessary. To be sure, communication between the agents had to be engineered (protocols established, data repackaged and reinterpreted, etc.), but MITRE readily acknowledges that they "have in reality engineered these systems to work together in an ad hoc manner; we have not

COPES: A Preliminary Analysis

achieved cooperation in the sense that the systems actually know semantically or from a knowledge-base standpoint about each other."

There are, though, important lessons, some obvious and some more subtle, to be learned from ALLIES that may prove to be of significant long-term value to COPES. Among the more obvious as those pointed out by MITRE: (1) that the partitioning of the problem so as to reflect how teams of humans perform the task has led to a workable, useful tool, (2) that using separate hardware systems has virtually eliminated competition for resources, (3) that the asynchrony of the systems reduces the required changes to the domain-specific parts of each system, and (4) that such "loose coupling" translates into a minimal commitment to change the existing experts and, thus, is readily extendable to other such integration efforts.

What is not spelled out clearly in the available literature is the reason why achieving a more comprehensive form of cooperation proved so difficult. The answer probably lies in the complexity of the individual systems themselves. As the individual experts to be integrated become themselves increasingly complex - both in terms of knowledge bases and algorithms - it is difficult to negotiate even basic communication. Problems arise that touch upon some of today's critical areas of AI research. How, for example, do distinct experts view and use the same data? Which data items are meaningful or available to which experts? How is global data maintained? updated? These are questions that occurred to us in specifying COPES, but for which there are no ready answers. The ALLIES project (and any planned follow-ons) may, under closer scrutiny, shed valuable light on them.

3.3 Teknowledge's ABE

ABE is a multi-level, general purpose software architecture intended to facilitate the design and implementation of large-scale AI applications and the reuse and integration of existing software components. At its base level (that most relevant to the COPES analysis), ABE can be considered a virtual machine (*Module Oriented Programming System. MOPS*) and cooperative operating system (*KIOSK*). MOPS describes a system as a set of modules communicating with each other by sending messages over networks. Modules may be primitive or may be recursively composed of smaller modules with their own communications requirements. A local controller manages the internal and external communication activities as well as the internal allocation of processing resources of each composite module. Communication between local controllers is accomplished by broadcasting messages over the connecting networks. Each local controller is responsible for broadcasting, filtering and distributing messages among its constituent modules.

This "modules-on-a-network" view is general in the sense that it may be used to describe a variety of distributed problem solving frameworks. A particular framework (e.g., a blackboard approach, or a dataflow framework) is implemented by building into a module's local coordinator design choices which specify control algorithms, resource allocation schemes, communications protocols, etc. In any case, the distinguishing features of an ABE-like network are (1) the use of local coordinators to manage independent groups of modules, and (2) these coordinators communicating via

COPES: A Preliminary Analysis

broadcast messages over a global network.

If we view COPES in its simplest form, as managing cooperation between two existing experts, an ABE-like network may in fact be overkill. It would hardly make sense in that case to resort to broadcast messages, or to alter directly a working expert's control algorithms or resource allocation schemes. In general, the idea that individual components must be aware of their role in a global problem solving scheme and, further, must have extensive knowledge of their relationship with all other modules in the network would also seem, as mentioned above, to impose significant change to each expert. On the other hand, isolating such information in a separate controller serves both to localize the information (and thus enhance the modifiability of the expert) and to impose a hierarchy on each constituent module. In the COPES scenario, such a hierarchy would allow for more explicit control over the individual experts.

3.4 USAICS' Multiple Node Expert System

The Multiple Node Expert System under development at Ft. Huachuca is a testbed environment for experimentation in cooperating expert systems. At present, nine existing systems have been identified (seven developed in house, the others developed by contractors) as contributing to the global problems of battle planning and battle management. The expert systems, running on separate machines linked by a local network, cooperate much as human specialists would in battle scenarios - i.e., by recognizing and responding to problems they are expert in, reporting their results to an overall commander, and assisting other experts as requested. The existing systems include experts for Commander's Guidance, Dissemination, Situation Development, Target Analysis, Imagery Analysis, ELINT Analysis, ELINT Correlation, HUMINT Analysis, and Requirements Management

The architecture and the technology underlying this cooperation is in fact theoretically primitive. As each system is booted, it polls the network to see which other experts are present. When an inference is made that is related to one of the other experts, that expert is invoked via remote function call from the expert that made the original inference, who in turn temporarily suspends its operation. The invoked expert acknowledges receipt of the message (so that the original expert can resume its processing), stores it, and processes it with its other current information. If and when the invoked expert has something to report back to the message originator, the same suspend/send/acknowledge/resume procedure is followed.

In order to accomplish such free-flowing communication, each expert must know a great deal about its fellow experts, including at a minimum how to invoke them, which of them may be of service in which situations, what their data formats are, etc. The resident experts are essentially "soft-wired" together. From the COPES perspective, this would necessitate substantial and non-trivial modifications to each of the cooperating experts. Given the priorities of this project though - foremost being the determination of information and processing requirements necessary to facilitate such cooperation - this straightforward soft-wire approach has great potential for experimentation, particularly as regards training and knowledge acquisition. The

COPES: A Preliminary Analysis

control strategy is, in principle if not implementation, ideal for full-fledged cooperation a la COPES.

4.0 Recommendations

There are aspects of each of these approaches that seem to apply to the COPES scenario. Whereas a blackboard facility addresses directly the need for sharing intermediate results and data, exploits opportunistic (and potentially parallel) control schemes, and also supports implicit communication between otherwise independent agents, a network configuration is more amenable to handling explicit tasking problems, hierarchical control algorithms and communication incompatibilities. Smith and Davis [13] refer to these complementary classes of cooperation as *result-sharing* and *task-sharing*, respectively.

Result-sharing is used as a means to enhance subproblem solution when such problems cannot be solved by a single, non-communicative agent. Our framework should facilitate this type of cooperation which allows the actions and results of one agent to influence those of another. Although task-sharing is used primarily to organize problem decomposition (which in the case of existing systems has already been accomplished), it works best for problem domains in which subproblems are more clearly independent, require minimal communication, and can be organized hierarchically - all of which would seem to pertain to existing systems being coordinated in the interest of attaining some high-level goal.

Smith and Davis foresaw the need for both types of cooperation and envisioned systems in which both forms were used. They proposed a task-sharing approach to tolerating uncertainty among agents (normally conceived of as a result-sharing problem), in which individual agents communicate their partial results, information about their goals, etc., to an external agent which serves as a coordinator. The coordinator has the option of retasking or reinvoking individual agents based on current information.

Use of an external coordinator imposes a hierarchical structure on the problem domain and is a standard organizational technique for dealing with problems of coordination. A hierarchical framework seems ideally suited to the case where the systems to be coordinated already exist. In such cases we do not have the luxury of being able to build into each agent the necessary facilities for cooperation as it is being implemented. (Although, an important outgrowth of the COPES effort may well be a more clear, generic specification as to how such facilities could be designed into future expert systems.) On the other hand, knowledge about which agents can perform and support which tasks and about what information can be usefully shared among agents - i.e., the reasons for coordination - are known a priori and can be incorporated into the coordinator. Our framework takes advantage of this fact and uses it as the basis for designing a cooperative system.

4.1 A Cooperative Framework

The coordinator has responsibility for tasking the individual agents and for

COPES: A Preliminary Analysis

facilitating implicit communication between them. It alone knows when and how distinct agents may be of service to one another. It directs both the flow of control and information through its interactions with the agents. Such interaction is accomplished via a controlled interface which specifies precisely the global information relevant to that particular agent. These interfaces are in effect blackboard-like models of the individual agents as they relate to the global task. They describe the partial results, data items and goals that are of global interest, along with a set of filters for encoding and decoding this information so that it may be used by the coordinator.

Such a framework combines aspects of each of the architectures evaluated in section 3, above. The coordinator operates more or less opportunistically based on its interpretations of the agents' models, which themselves are mini-blackboards. These models serve a similar purpose to the "local controllers" prescribed by Corkhill and Lesser for organizational structuring, but are unencumbered by extensive global knowledge. The proposed framework is pictured in Figure 1, below.

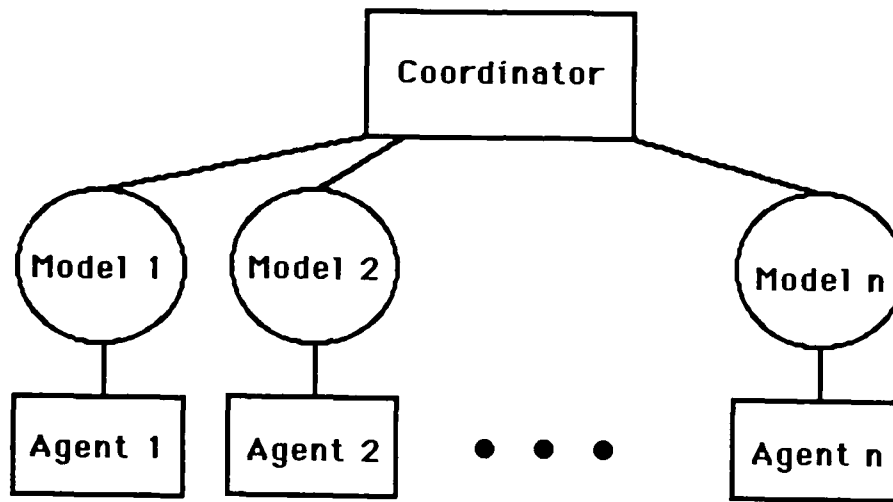


Figure 1

This framework satisfies our criteria for coordinating previously-defined agents. It minimizes the need for changing the agents by freeing them from having to know too much about either the high-level concerns of the coordinator or the roles of other agents in the system. This autonomy allows for unlimited expandability - i.e., new agents can be added to the system by changing only those models and agents which have the potential for interacting with the new agents. Similarly, new high-level tasks can be taken on by the coordinator and only those agents which support that task need be changed. Changes are unavoidable for knowledge-based agents (other, more conventional agents which do not have the prospects of benefiting from the results or actions of others would not require modification), but because of the framework these changes are also localized and clearly specified.

Each agent must be modified in two general ways: (1) it must post and update the information prescribed by its model so that it is available to the coordinator, and (2) it

COPES: A Preliminary Analysis

must have the potential for suspending and resuming its operation, a la the Multiple Node Expert System, at points prescribed as goals of global interest by its model. Thus, the only assumptions we need to make about knowledge-based agents are that they be amenable to maintaining a simple data base of information for global access and that they be capable of articulating and posting their plans and goals.

The hierarchical nature of the framework also makes sense from a software engineering point of view. A hierarchy simultaneously eliminates the connection problem (i.e., having to endow individual agents with information about what and how they will communicate with other agents), localizes and makes explicit the changes required of agents, and, by embedding all global control information in the coordinator, insures that the scope of effect for major decisions is within the coordinator's scope of control.

4.2 A Design Technique

The proposed framework is applicable to the task of coordinating existing experts in large part because the reasons for coordination are known in advance of developing the system. The task of designing the coordinator and the agent models is primarily one of data specification - i.e., we must specify the information content and the associated flow of information for each. Many hierarchical expert systems are described in terms of successive levels of abstraction on the data involved. A technique for designing a coordinated system would ideally facilitate the process of transforming reasons for coordination into the data items that make up the agents' models and drive the coordinator.

Basili's top-down data collection methodology [1] describes such a general technique. Data collection is viewed as being dependent on a clear description of the reasons for collection, and should provide a means for linking these reasons with the data items they relate to. The methodology specifies that the reasons for collection be used to generate a set of questions which quantify the reasons - i.e., articulate criteria by which they can be attained. These questions, in turn, serve to identify the specific data items relevant to the original goals.

In the context of our framework, this top-down process can be interpreted as follows:

- (1) specify the reasons for coordination - i.e., the potential interrelations among the existing agents (expert and non-experts),
- (2) on an agent-by-agent basis, identify the information necessary (both goal- and data-related) to support these interrelations, and
- (3) on an agent-by agent basis, describe the changes required of the agent in order for it to provide the information specified in (2), and for it to suspend and resume processing when such information is goal-related.

The reasons specified in (1) serve to describe the control structure of the coordinator. The information identified in (2) describes each agent's model. The changes described in (3) are precisely those that must be imposed on the input/output and control

COPES: A Preliminary Analysis

structures of the individual agents. A brief hypothetical example will simultaneously illustrate the utility of this technique and free us from worrying about the implementation details of existing expert systems.

4.3 An Example

Imagine two systems which help to plan the respective activities of a small town's fire (FD) and police departments (PD). Each is fully operational and is capable of planning, scheduling and monitoring the activities of its personnel - i.e., each is an expert system in the current sense of the term. A community services coordinator (CSC) is to be designed which makes use of the existing systems and coordinates their activities. The need for such a coordinator stems from the recognition that the existing systems have a potential for interaction in a variety of ways. That is, the reasons for coordinating the systems are known from the outset to include:

- (1) the PD can help to control traffic and thereby support the FD in planning routes to a fire and in planning how to attack a fire,
- (2) the FD can assess a bomb threat for potential fire damage and thereby support the PD in planning how to handle a threat situation,
- (3) the FD can detect the possibility of arson and must notify the PD, and
- (4) the PD can detect a fire and must notify the FD.

Clearly, there are numerous other interactions between real agents, but these serve to illustrate the two general classes of cooperation we spoke of earlier.

Cases (3) and (4) are instances of mere communication between agents and can be handled as a task-sharing problem by the CSC. Cases (1) and (2) involve a decidedly more complex form of cooperation wherein one agent can influence another during processing. In order to accomplish this result-sharing, the CSC must be aware of the dynamic state of the agents and must make the intermediate results from one available to the other. Our framework and the accompanying design technique help us to describe a system, in Figure 2, which facilitates both forms of cooperation and imposes a minimum of controlled change on the FD and PD systems.

The reasons articulated above serve as guidelines for the interface models that must be projected by each of these experts. For example, reason (1) dictates that the coordinator be alerted whenever the FD expert undertakes a goal involving the need for traffic control (either route-planning or deciding how best to attack a fire). Similarly, reason (2) dictates that the coordinator be alerted whenever the PD is called upon to assess a bomb threat. These are the two recognized cases where the previously-independent agents have the potential for influencing each others' actions. Each model has areas reserved for the posting of these goals and both models have space reserved to hold the global data values related to them (State_of_traffic and Fire_potential). To accomodate reason (3), the FD's plan for evaluating the likelihood of arson shows up in its model as a posted goal and matches the data area for arson information (Arson_detected) in the PD model. The PD plan for checking for fire conversely matches the Fire_detected data area in the FD model to accomodate reason (4).

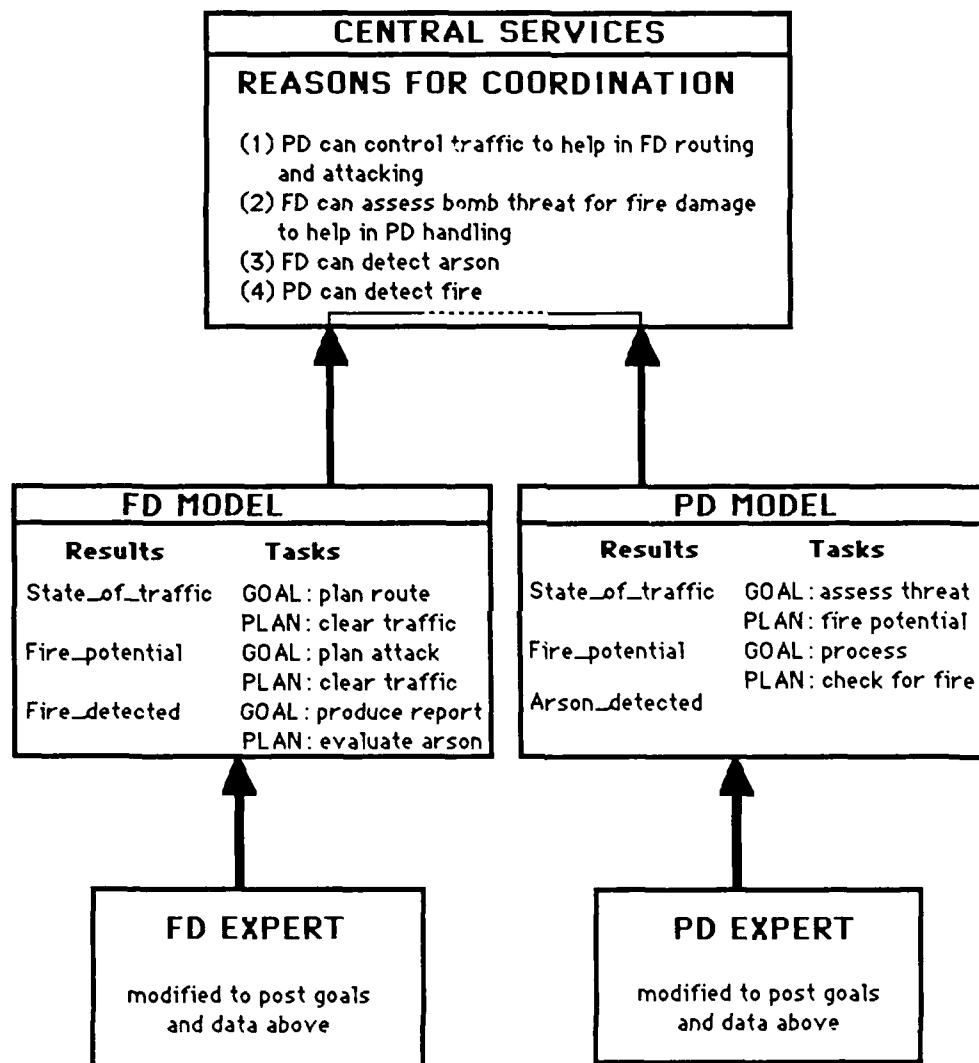


Figure 2

COPES: A Preliminary Analysis

Each expert must be modified in order to support its projected model. Specifically, each would be modified to allow it to post and maintain the goals and data values in its model. Equally important, for each goal posted, the agent must be capable of suspending (not terminating) its operation so that the coordinator has the opportunity to intervene and respond to the goal in an appropriate way - i.e., by invoking the other expert.

Imagine, for example, the the FD expert is invoked to respond to a fire. It establishes as a goal the planning of a route to the fire. In order to accomplish this, it must take into account traffic and in so doing, posts its intentions and suspends operations. Recognizing this, the coordinator invokes the PD expert with a request to clear traffic along the considered route. The PD processes this request and posts its results on its model. The coordinator reads the results, transmits them to the FD model and reinvokes the FD expert to resume processing by reconsidering its most recent goal (route planning).

If, on the other hand, the PD was invoked to respond to a bomb threat, it would as part of its response establish the goal of evaluating the potential for fire danger. This goal would be posted in its model, recognized by the coordinator, and the PD would suspend its operations. Upon assuming control, the CSC would invoke the FD expert to evaluate the fire hazard. The FD would post its results and the CSC, in turn, would transmit them to the PD, reinvoking it in the process.

The control structure of the coordinator, CSC, is opportunistic in that it is built to recognize precisely those goals and data values having global significance. Whenever the individual models are altered (thus indicating a need for global action), the coordinator assumes control. The sequence of invocations and postings that follow are, in each case, predetermined, again, by the high-level reasons for coordination.

4.4 Conclusions

In summary, the task of coordinating the activities of existing expert systems is seen as highly constrained when compared with the case of building a coordinated system from scratch. At first glance, the use of existing systems would seem to necessitate making a significant tradeoff between the extent of the modifications that must be made and the extent that such systems can be made to cooperate. Upon closer examination, the fact that the designer does not have unlimited freedom in designing the experts so as to facilitate communication (i.e., providing them with common architectures, data structures and formats, and communication protocols) is nearly offset by the fact that the reasons for coordinating the existing systems are known in advance.

These reasons, when carefully articulated, can be used to describe (1) the control structure of a high-level coordinator, (2) the interface that each of the individual agents must project to the coordinator, and (3) the changes required of each agent. The proposed hierarchical framework and corresponding design technique appear to minimize and effectively localize the required changes while allowing for a full range of

COPES: A Preliminary Analysis

cooperative problem solving behavior.

COPES: A Preliminary Analysis

5.0 Bibliography

The references listed below are those that were most directly useful in preparing this report. Clearly, numerous other papers, reports, texts and company proprietary documents also provide supportive material for the wide ranging issues addressed. What follows is a representative subset.

1. Aiello, N. (1986). "User-Directed Control of Parallelism; The CAGE System," Technical Report, Knowledge Systems Laboratory, Computer Science Department, Stanford University.
2. Basili, V., S. Hirshfield and R. Iuorno. (1985). "Determination of Data Collection Needs for the Ada Program: A Methodology for Identification and Collection," Technical Report, IIT Research Institute.
3. Benoit, J.W., R.P. Bonasso, J.R. Davidson, J.C. Ellenbogen, S.J. Laskowski, R.W. Wong. (1986). "Airland Loosely Integrated Expert Systems: The ALLIES Project," MTR-86W00041, The MITRE Corporation.
4. Bonasso, R.P. (1984). "Expert Systems for Intelligence Fusion," *Proceedings of the Army Conference on Application of Artificial Intelligence to Battlefield Information Management*, AD-A139-685, 101-106.
5. Buchanan, B.G. (1981). "Research on Expert Systems," STAN-CS-81-837, Stanford University.
6. Cammarata, S., D. McArthur, and R. Steeb. (1983). "Strategies of Cooperation in Distributed Problem Solving," *Proceedings of the IJCAI 1983*, 767-770.
7. Clancy, W.J. (1983). "Advantages of Abstract Control Knowledge in Expert System Design," STAN-CS-83-995, Stanford University.
8. Cooke, N.M. (1985). "Modelling Human Expertise in Expert Systems," MCCS-85-12, Computing Research Laboratory, New Mexico State University, Las Cruces, NM.
9. Erman, L.D., M. Fehling, S. Forrest, and J.S. Lark. (1986). "ABE: Architectural Overview," notes for presentation at Workshop on Distributed Artificial Intelligence.
10. Erman, L.D., J.S. Lark, and F. Hayes-Roth. (1986). "Engineering Intelligent Systems: Progress Report on ABE," TTR-ISE-86-102, Teknowledge, Inc.
11. Hamill, B. (1984). "Psychological Issues in the Design of Expert Systems," Technical Report, Milton S. Eisenhower Research Center.
12. Hayes-Roth, B. (1983). "The Blackboard Architecture: A General Framework for Problem Solving?," HPP-83-30, Heuristic Programming Project, Computer Science Department, Stanford University.

COPES: A Preliminary Analysis

13. Hayes-Roth, F., D.J. Mostow, and M.S. Fox. (1978). "Understanding Speech in the HEARSAY-II System," in L. Bolc (ed.) *Speech Communication With Computers*. Munich: Carl Hansen Verlag..
14. Hayes-Roth, F., D.A. Waterman, and D. Lenat (eds.). (1983). *Building Expert Systems*. Reading, MA: Addison-Wesley.
15. Kline, T.J. and S.B. Dolin. (1985). "Choosing Architectures for Expert Systems," RADC-TR-85-192, Rome Air Development Center.
16. Lesser, V.R. and D.D. Corkhill. (1981). "Functionally Accurate Cooperative Distributive Systems," *IEEE Transactions on Man, Systems and Cybernetics*, SMC-11, 81-96.
17. Lesser, V.R., R.D. Fennel, L.D. Erman, D.R. Reddy. (1974). "Organization of the HEARSAY-II Speech Understanding System," in *IEEE Symposium on Speech Recognition*. 11-M2-21-M2.
18. Morris, J. (1985). "Application of Tools, Technologies and Methodologies for Rule-Based Expert Systems," Technical Report, IIT Research Institute.
19. Nii, H.P. (1986). "Blackboard Systems: The Blackboard Model of Problem Solving and the Evolution of Blackboard Architectures," *The AI Magazine*, Summer, 1986, 38-53.
20. Nii, H.P. (1986). "CAGE and POLIGON: Two Frameworks for Blackboard-based Concurrent Problem Solving," Technical Report, Knowledge Systems Laboratory, Computer Science Department, Stanford University.
21. Rice, J.P. (1986). "POLIGON, A System for Parallel Problem Solving," Technical Report, Knowledge Systems Laboratory, Computer Science Department, Stanford University.
22. Robbins, J. (1986). "The Application of Artificial Intelligence Techniques to the Processes of Military Intelligence; A Problem Driven Approach," Technical Report, Artificial Intelligence Team, U.S. Army Intelligence Center and School, Ft. Huachuca, AZ.
23. Smith, R.G. and R. Davis. (1981). "Frameworks for Cooperation in Distributed Problem Solving," *IEEE Transactions on Man, Systems and Cybernetics*, SMC-11, 61-69.